Computer Science Department

TECHNICAL REPORT

Automated Assembly of a Jig-Saw Puzzle Using the IBM 7565 Robot

by

Grigore Burdea
Haim Wolfson

Technical Report No. 188
Robotics Report No. 55
November 1985

NEW YORK UNIVERSITY

Department of Computer Science
Courant Institute of Mathematical Sciences
251 MERCER STREET, NEW YORK, N.Y. 10012

[redacted]

Automated Assembly of a Jig-Saw Puzzle Using the IBM 7565 Robot

by

Grigore Burdea
Haim Wolfson

Technical Report No. 188
Robotics Report No. 55
November 1985

New York University
Dept. of Computer Science
Courant Institute of Mathematical Sciences
251 Mercer Street
New York, New York 10012

Work on this paper has been supported by Office of Naval Research Grant N00014-82-K-0381, National Science Foundation CER Grant DCR-83-20085, and by grants from the Digital Equipment Corporation and the IBM Corporation.

AUTOMATED ASSEMBLY OF A JIG-SAW PUZZLE USING THE IBM 7565 ROBOT[1]

Grigore Burdea[2] *and Haim Wolfson*

Robotics Research Group
Courant Institute of Mathematical Science
New York University

ABSTRACT

This preliminary paper describes some of the aspects of automated jig-saw puzzle assembly. The Robotics Group at N.Y.U. has recently developed an algorithm for vision recognition and software assembly of a jig-saw puzzle. We present some other steps to be taken in order to integrate the present knowledge in a complete assembly of the puzzle parts by a robot.

1. Introduction

Recent research done by Edith Schonberg and Haim Wolfson at N.Y.U.'s Robotics Laboratory has led to the almost complete development of an integrated process of vision recognition and software assembly of a jig-saw puzzle.

---

[1] Work on this paper has been supported by Office of Naval Research Grant N00014-82-K-0381, National Science Foundation CER Grant DCR-83-20085, and by grants from Digital Equipment Corporation and the IBM Corporation.

[2] Affiliated with Department of Applied Science, New York University.

The process consists of three major steps. First an RCA 2000 black and white camera is used to take pictures of the jig-saw puzzle pieces and feed information about the contour of the pieces into a VICOM image processor (in order to be digitized). This process is repeated until all pieces of the puzzle are stored into the memory of a VAX 750 host computer to form a data bank. The second step is to apply a local boundary matching algorithm which produces scores for the matchings of the puzzle pieces with each other. Finally a global algorithm based on combinatorial optimization techniques is applied in order to obtain the puzzle assembly. The last two steps are closely related as described in Section 2.2 *Integrated software-hardware assembly*. Experiments are presently done on 100-piece puzzles and work is in progress to make the software algorithm real time (See Fig. 1, *Early Smaller Puzzle*).

In the next section we describe an interactive procedure where a robot participates in the puzzle assembly by receiving partial solutions from the software algorithm, checking them, sending the results back to the algorithm where this feedback is used to improve the solution. We estimate that after a small number of such cycles the computer will supply the robot with the exact solution which the robot will then assemble.

2. Automated assembly of a jig-saw puzzle

The automated assembly of a jig-saw puzzle combines two processes. The first is visual recognition of the puzzle pieces, which is done in order to obtain data related to the shapes of the pieces that belong to a given puzzle.

The second process is the software-hardware assembly, which results in the complete and correct assembly of the puzzle.

In our approach we start with the assumption that the robot has no previous knowledge about the puzzle and that no information related to the assembled puzzle picture had been furnished.

2.1. Visual recognition

The pieces can very well be left in a pile on the robot table and are then retrieved one by one by the robot end effector. However, in order to pick a random piece from the pile, the robot might need an additional range sensor. We therefore intend to use a different technique in which the camera "scans" the field. First all the puzzle pieces have to be displayed on a flat surface and the camera mounted on the robot arm. The view field of the camera can be adjusted so that the total robot table surface is a multiple of the strip scanned by the camera on one lateral scan. The possible cases in terms of position of pieces within the camera field are presented in Fig. 2. In order to assure a proper illumination of the camera field, we suggest the use of a large opaque plastic sheet, with light sources placed under it and the puzzle on top of it (light table). The camera will then slide continuously over the field, recording all inputs and eventually reconstructing the individual pieces. Once this is done, each piece should have an ID which includes its center of gravity coordinates X and Y in the puzzle field. Such data is very useful later when the robot tries to pick up the particular piece in the field which the computer

has determined is next to be assembled. A robot which could support such a scanning of a puzzle field, and which has X and Y Cartesian coordinates, is the IBM 7565. One could therefore construct the above-mentioned opaque illuminated plastic support and mount it on the robot table. Care has to be taken to assure a very good parallelism between the robot table and the light table. The video camera could be mounted on the robot arm and thus be free to move above the puzzle field and focus on any X and Y location. When mounting the camera on the robot arm, we must interfere as little as possible with the work envelope of the robot. It is therefore useful to use a camera that has small dimensions and thus has little effect on the robot movements in the Z coordinate. One solution for these problems is the use of a CCD 3000 Video Communications Camera (Fairchild), which could have the sense head mounted on the robot arm and be connected through a flexible cable to the control unit [2].

In order to increase accuracy in the data input, we could use lenses that limit the camera's angle of view to 10-15 degrees. A circle of 5" diameter should be sufficient because it corresponds to roughly twice the dimension of a puzzle piece. If this field is scanned by a camera that produces 512 elements per line (best case), then the discretization obtained is of the order of .007" or worse. The IBM 7565 has a repeatability in the X and Y coordinates of .004-.005" which reduces the problems of image errors due to inaccurate arm/camera positioning. Once a particular piece is "discovered" in

the strip scanned by the camera, the robot arm/camera assembly can move to place the puzzle piece in the center of view. Then the camera can stop and take the picture so that the effect of vibrations is reduced. This is the simple case (a) (see Fig. 2). In the case (b), only part of the piece falls within the first strip, so that the camera has to leave the parallel path in order to get a complete view of the piece. After this, the picture is taken and the center of gravity coordinates are recorded; thus the camera will not input the data for the same piece twice when the next scanning pass is done. In case (c), when the pieces overlap, the robot has to detect the resulting larger area and then separate the pieces using the end effector, after which the whole moving and digitizing process is repeated. The software should assure a minimum 1/8" space between the edges of the pieces that have been separated by the end effector, so that no errors occur in the data input.

2.2. Integrated software-hardware assembly

In this section we describe briefly the existing software algorithm and how we envision the future interaction between it and the robot. As we mentioned in Section 1, *Introduction*, the software algorithm consists of three major steps. The first one is concerned with data acquisition and digitization of the boundary curves of the puzzle pieces. The second one computes scores for matchings between different pieces, using the Schwartz-Sharir curve matching algorithm [1]. The third step is a global algorithm which uses the

information obtained in the previous steps in order to assemble the puzzle. We see a close interaction between the software and the hardware and, hence, we describe it in more detail. This part of the algorithm has two phases.

1) Assembly of the perimeter of the puzzle.

This phase starts with recognition of the perimeter pieces based on the fact that they have special line segments in their boundary (e.g. straight lines for square puzzles or circle arcs for circular puzzles or any other specifications). Recognition is based on the visual data but can also be accomplished, or at least checked, by the robot. Once the perimeter pieces are known, the basic matching algorithm (Phase 2) supplies us with a matching matrix of every piece against all other pieces. Then a combinatorial optimization algorithm, the so called "Traveling Salesman" (T-S) algorithm, is applied to get the optimal global match of all the perimeter pieces. The results obtained using this algorithm will be supplied to the robot in order to assemble the perimeter and check the software proposition. If all the matchings are correct the algorithm will pass to Phase 2. If not all the matchings are correct the robot will feed back data about correct, incorrect and undecidable matches from the robot's point of view. The software will use this information to alter the basic matrix and will run the T-S algorithm again. This kind of feedback is expected to have a significant impact on the overall performance of the algorithm and we project that no more than 4 or

5 iterations will be needed. Once the perimeter (frame) is solved and assembled we pass to Phase 2.

2) **Assembly of the inner pieces of the puzzle.**

The second phase is a dynamic programming algorithm, which uses the fact that once the frame is arranged, we get "corners" where a puzzle piece has to match with about half of its boundary line. The algorithm tries to match pieces into these corners and stores a number of best results for the next step. Once a piece is selected to fill a corner, new "corners" result and the algorithm proceeds. Here again we see a possibility of close interaction with the robot. It can be done locally -- the robot can actually try to put a number of "best matching" pieces into a corner and feed back the results of these trials. Hopefully one of the pieces will match and the software algorithm will proceed to another corner. The interaction could be done in a global manner as in Phase 1. The robot will accept a global assembly proposal from the software, will try to put all the pieces together and will feed back the results to the software algorithm for the next iteration. This approach seems to be less time-consuming, provided the software algorithm has a good performance. An intermediate approach is also possible here. The software algorithm will stop after a prescribed number of steps, or after it realizes that the overall results are getting worse, and will send the partial solution to the robot for a hardware check. Then the algorithm proceeds from a new solid starting point. The exact approach will be based on real-

time considerations and the quantity and quality of input data. At the end of Phase 2 of the interactive algorithm we will have an assembled puzzle.

3. Hardware considerations

The automated assembly of a jig-saw puzzle is a complex process from the robot point of view and it is not a simple pick-and-place routine. The entire process involved in getting a correct puzzle assembly requires many steps and is subject to numerous potential errors. Specifically, pieces of complex shape which are made out of cardboard, could be easily deformed due to repeated handling by the robot end effector. We suggest that initially we would use wood pieces that will better withstand the forces applied by the gripper. Once the algorithms were developed and tested, we could switch to lighter materials such as cardboard.

The assembly can be done on one half of the robot table while the pieces are picked from the other half (See Fig. 3). The puzzle hardware assembly begins with the selection of the part determined by the computer to be the one that matches the previous mounted piece. The selected part could then be gripped by its lateral edges from the top and bottom. Use of a suction device would secure its top only, which seems to be the best solution in terms of work envelope and clear view of the work area for the camera. Once the part is gripped, it is moved to the desired location in the assembled puzzle and put into place. Here the algorithm should perform several levels of checks to ensure that the part has been properly mounted (Fig 4). The first

check is done during the attempt to put the part in place, when pressure sensors monitor the pressure applied on the arm (in response to the arm movement). If this value is bigger than a certain threshold, it is most probable that there is some overlapping in the boundaries of adjacent pieces. At this point we will probably use some kind of tactile sensing in order to assure a proper assembly. Once the piece has been assembled, a second check is done to verify a smooth transition on the portion of the edge of the puzzle piece that corresponds to the previously mounted piece. This is done by having the touch sensors mounted on the robot end effector sweep the boundary between the two pieces in a continuous Z-shaped motion. If no gaps are detected, the level of confidence in the assembly step increases. In the case of gaps between pieces (Fig. 4(c)) some light from the light table will shine through and can be detected by the camera mounted on the robot arm. Any such patch of light could be detected as being a strongly lit closed contour surrounded by dark spaces (due to the interference of the puzzle piece with the light path). Another check that can be applied to "frame" pieces of a square puzzle is to (visually) determine the angle made by the two straight lines which are the external boundaries of two assembled pieces. If this angle is different from 180 degrees then the assembly is clearly incorrect.

The automated assembly of the jig-saw puzzle assumes that the robot has some advanced sensory feed-back. The robot cannot rely only on inputs from the controller (open loop) even if we consider the video camera for

feedback. Indeed the camera can be useful for determining approximate position and direction, but when it comes to fine positioning we need two things: a good touch sensor on the gripper fingers and some kind of compliance to allow for the inaccuracy in the assembly process.

The IBM 7565 robot gripper has some built-in sensory features which include a force-sensor that monitors the force applied by/on the gripper, and a proximity sensor that determines if some object is within the fingers of the gripper. This proximity sensor has a photodiode that monitors a light beam emitted by LED. Both are mounted on the gripper and signal "on" in normal position, i.e. when no object is detected between the fingers. This signal is turned off when some object intersects the light beam. The force sensor is of the load gauge type and the force signal is translated into resistance change [3].

We have shown that the automated assembly of the jig-saw puzzle requires very good sensory feedback. It seems that the built in features of the IBM gripper do not satisfy our requirements. We therefore have to develop new sensors and associated software.

4. Necessary future research

In view of the aspects previously discussed we think that research has to be undertaken in the following areas:

(a) A study of human movements related to a jig-saw puzzle assembly (picking and securing a piece, putting it in place, checking for correctness of assembly, etc.).

(b) Software development for scanning the pieces on the robot table for shape data input.

(c) An experiment for the assembly of two puzzle pieces which will also include overlap detection and removal software.

(d) Software development of an interactive assembly program that will accept feedback from the field to correct for errors in the selection of best shape fit.

(e) Development of software for the checking of the piece assembly by the robot.

(f) Hardware development of the end effector mechanics and sensory system. Building a light table on top of the robot table and installing a video camera on the robot arm.

(g) An experiment to digitize images of puzzle pieces with the camera mounted on the robot arm and to compare these results with data obtained using an immobile camera.

5. References

[1] J.T.Schwartz & M.Sharir, *Identification of Partially Obscured Objects in Two Dimensions by Matching Of Noisy 'Characteristic Curves'* , C.I.M.S. Robotics Report No.46, New York University, June 1985.

[2] CCD 3000 Video Communications, *Camera Operating Manual*, Fairchild, March 1983.

[3] IBM 7565 Manufacturing System Hardware Library, Publication No. 8508986, *Maintenance Information*, September 1983.

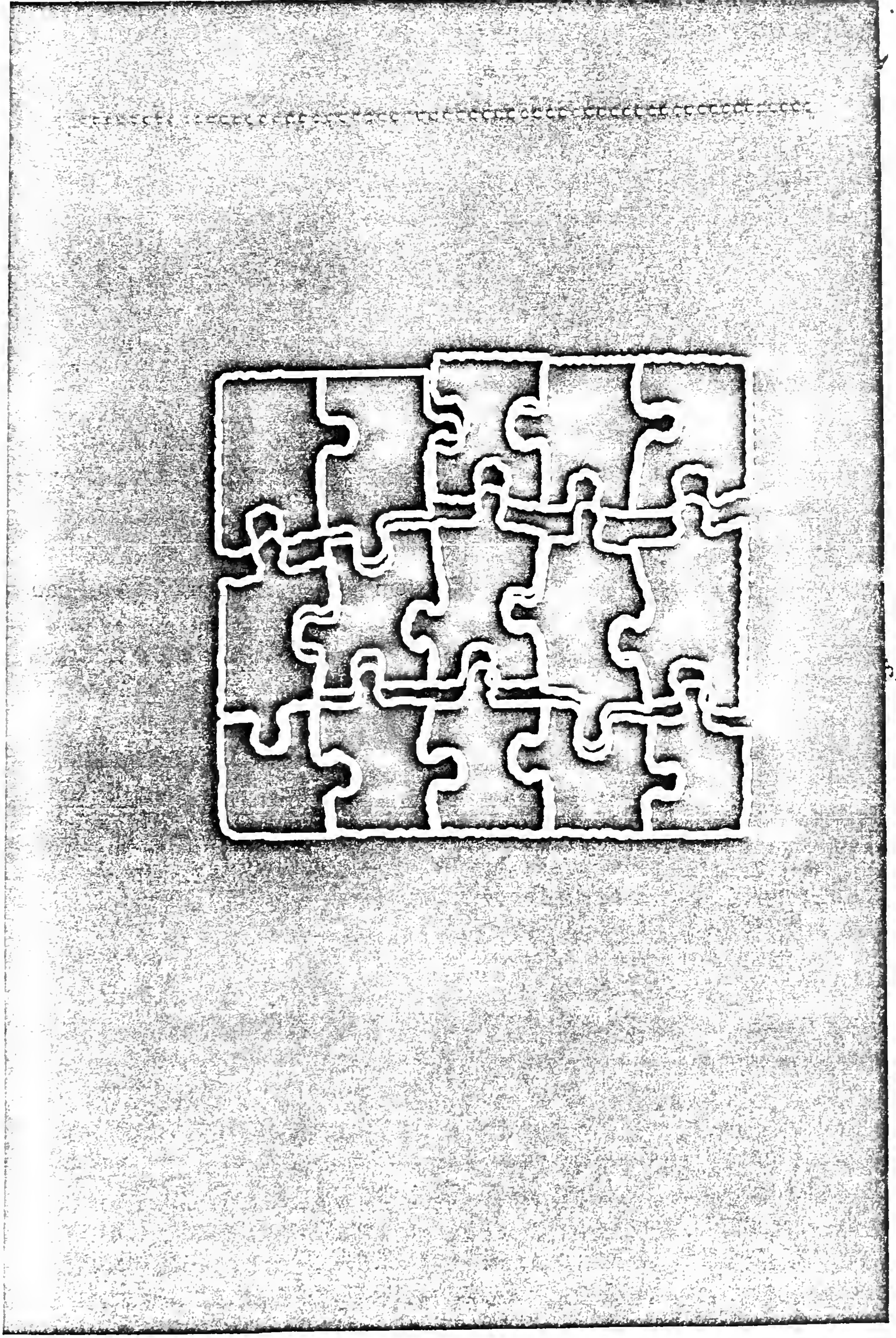

FIG. 1 EARLIER SMALLER PUZZLE

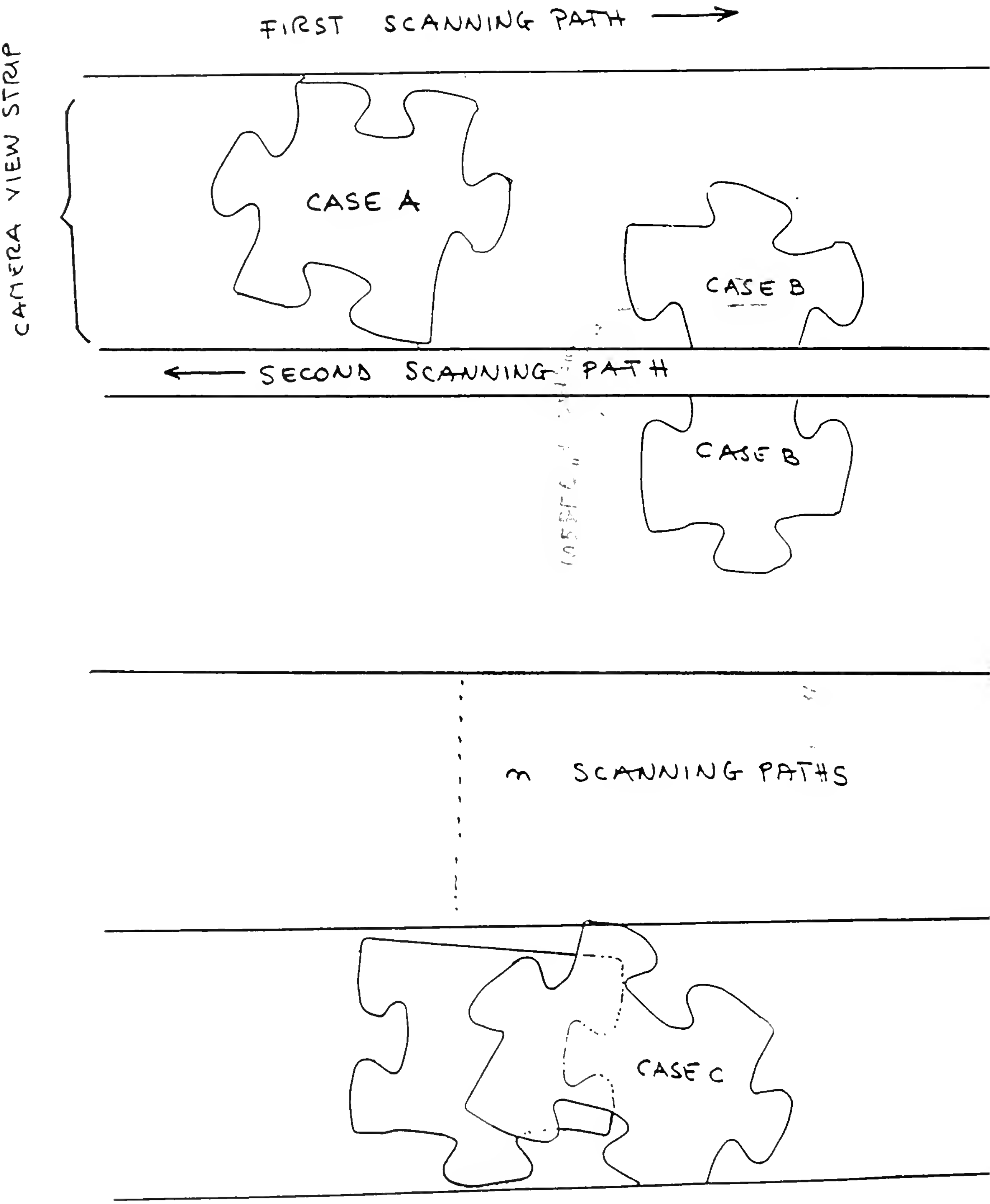

FIG. 2 PUZZLE PIECES IN CAMERA VIEW FIELD

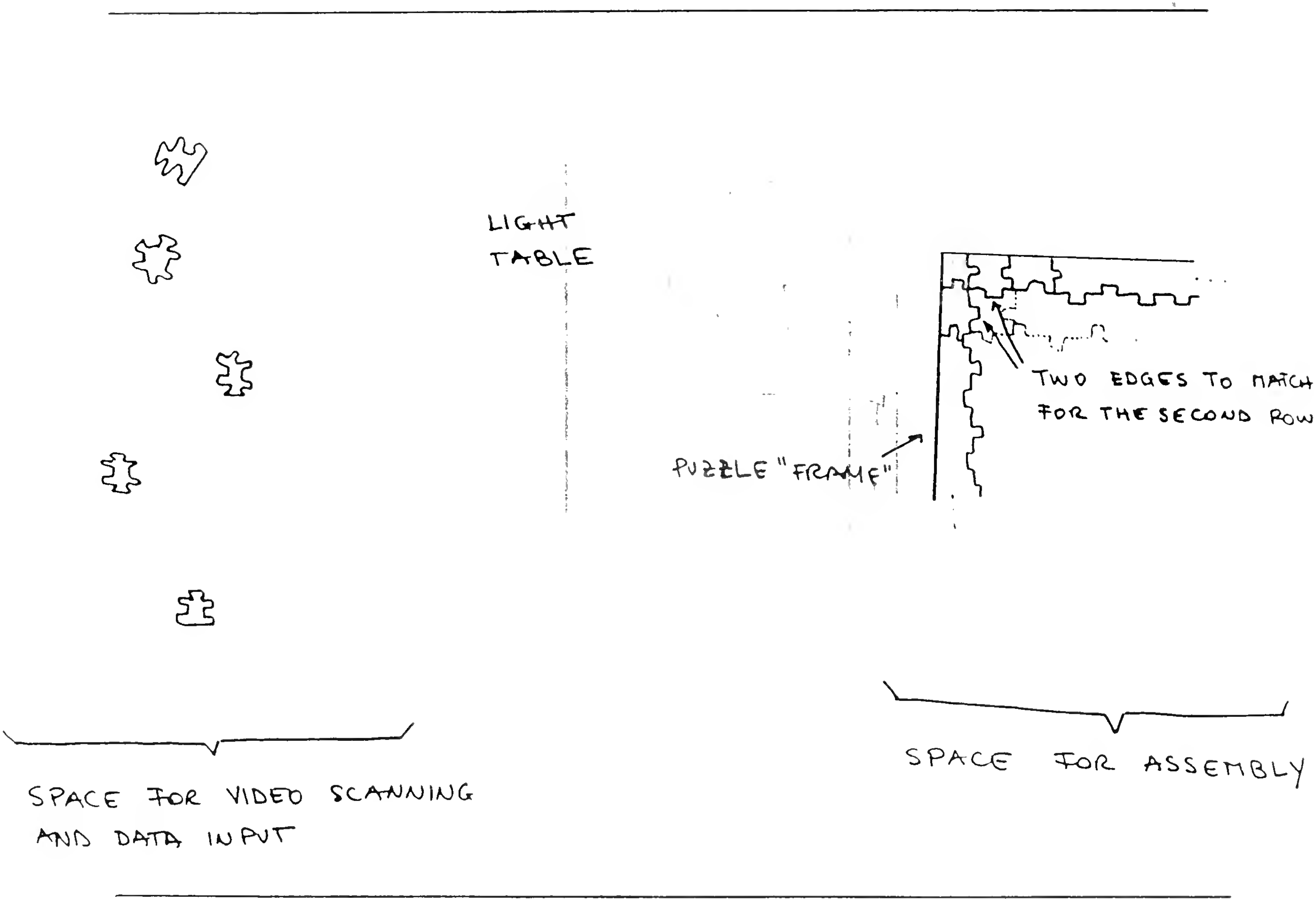

FIG. 3 PARTIAL PUZZLE ASSEMBLY ON ROBOT TABLE

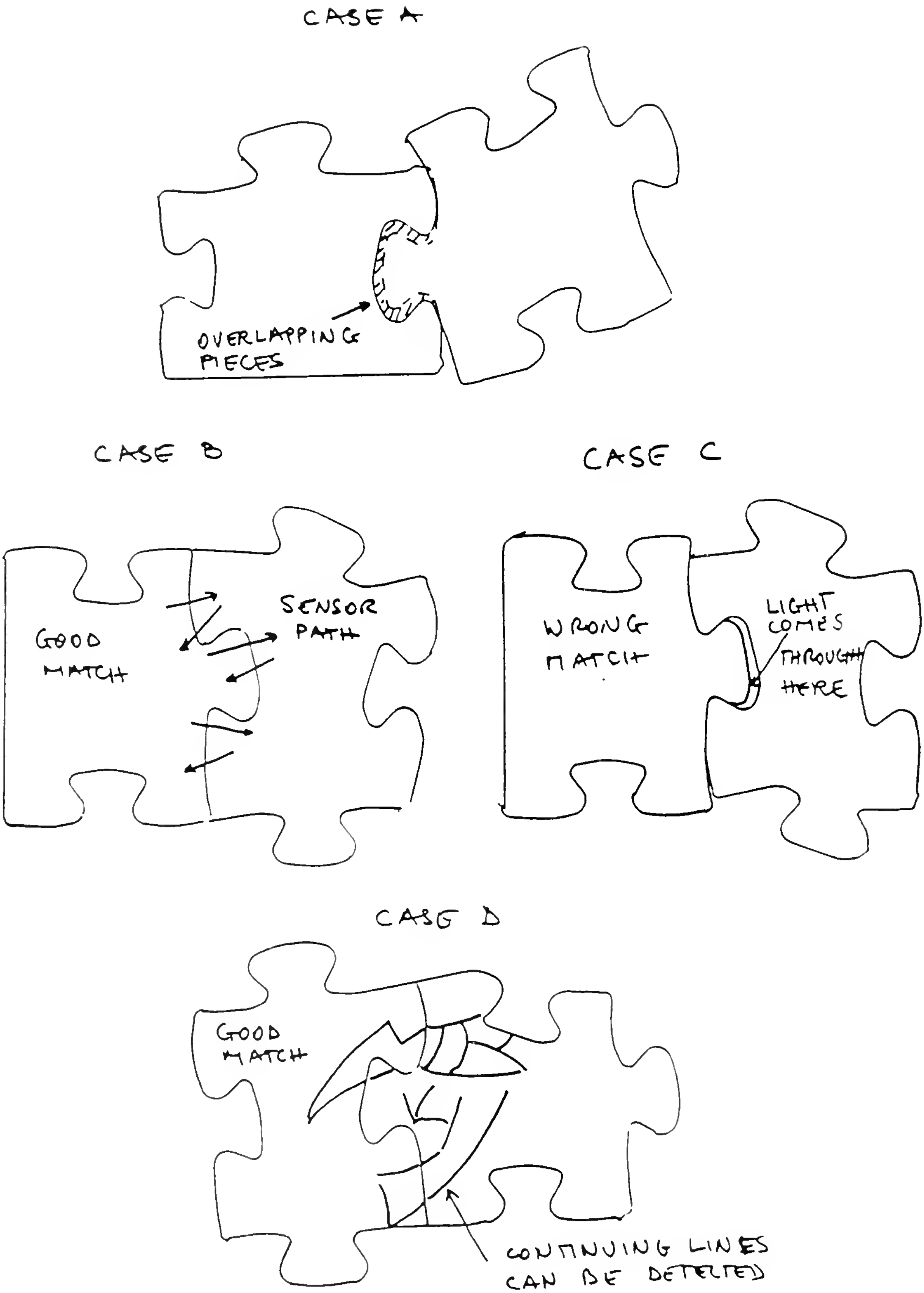

FIG. 4 ROBOT CHECKS OF CORRECT PUZZLE ASSEMBLY

NYU COMPSCI TR-188 c.1
Burdea, Grigore
Automated assembly of a
jig-saw puzzle using the
IBM 7565 robot